Tokenization/Segmentation

IP notices: slides from D. Jurafsy, C. Manning and S. Batzoglou

Outline

- Tokenization
  - Word Tokenization
  - Normalization
    - Lemmatization and stemming
  - Sentence Tokenization
- Minimum Edit Distance
  - Levenshtein distance
  - Needleman-Wunsch
  - Smith-Waterman

Tokenization

- For
  - Information retrieval
  - Information extraction (detecting named entities, etc.)
  - Spell-checking
- 3 tasks
  - Segmenting/tokenizing words in running text
  - Normalizing word formats
  - Segmenting sentences in running text
- Why not just periods and white-space?
  - `Mr. Sherwood said reaction to Sea Containers' proposal has been "very positive." In New York Stock Exchange composite trading yesterday, Sea Containers closed at $62.625, up 62.5 cents.`
  - `"I said, 'what're you? Crazy?' " said Sadowsky. "I can't afford to do that.''`

What's a word?

- I do uh main- mainly business data processing
  - Fragments
  - Filled pauses
- Are cat and cats the same word?
- Some terminology
  - **Lemma**: a set of lexical forms having the same stem, major part of speech, and rough word sense
    - Cat and cats = same lemma
  - **Wordform**: the full inflected surface form.
    - Cat and cats = different wordforms
  - Token/Type

How many words?

- they lay back on the San Francisco grass and looked at the stars
  - 13 tokens (or 12)
  - 12 types (or 11)
- The Switchboard corpus of American telephone conversation:
  - 2.4 million wordform tokens
  - ~20,000 wordform types
- Brown et al (1992) large corpus of text
  - 583 million wordform tokens
  - 293,181 wordform types
- Shakespeare:
  - 884,647 wordform tokens
  - 31,534 wordform types
- Let N = number of tokens, V = vocabulary = number of types
- General wisdom: V > O(sqrt(N))

Issues in Tokenization

- ***Finland's capital →***
  Finland? Finlands? Finland's
- ***what're, I'm, isn't->***
  - ***What are, I am, is not***
- ***Hewlett-Packard →***
    - ***Hewlett*** and ***Packard*** as two tokens?
  - ***state-of-the-art***:
    - Break up?
  - ***lowercase***, ***lower-case***, ***lower case*** ?
- ***San Francisco, New York***: one token or two?
- Words with punctuation
  - **m.p.h., PhD.**

Slide from Chris Manning

Tokenization: language issues

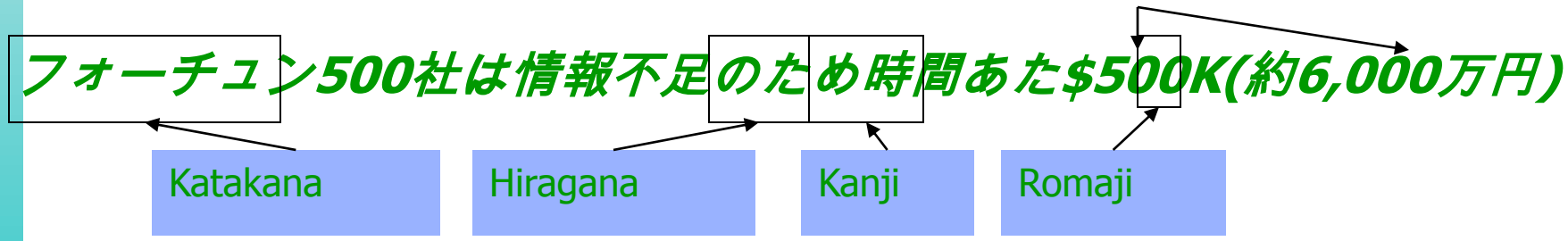
- Italian
  - ***L'insieme*** → one token or two?
    - ***L*** ? ***L'*** ? ***Lo*** ?
    - Want ***l'insieme*** to match with ***un insieme***

- German noun compounds are not segmented
  - ***Lebensversicherungsgesellschaftsangestellter***
  - 'life insurance company employee'
  - German retrieval systems benefit greatly from a **compound splitter** module

Slide from Chris Manning

Tokenization: language issues

- Chinese and Japanese no spaces between words:
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
  - Sharapova now lives in US southeastern Florida
- Further complicated in Japanese, with multiple alphabets intermingled
  - Dates/amounts in multiple formats



End-user can express query entirely in hiragana!

Slide from Chris Manning

Word Segmentation in Chinese

- Words composed of characters
- Characters are generally 1 syllable and 1 morpheme.
- Average word is 2.4 characters long.
- Standard segmentation algorithm:
  - Maximum Matching
    - (also called Greedy)

Maximum Matching Word Segmentation Algorithm

- Given a wordlist of Chinese, and a string.

1) Start a pointer at the beginning of the string
2) Find the longest word in dictionary that matches the string starting at pointer
3) Move the pointer over the word in string
4) Go to 2

English failure example (Palmer 00)

- the table down there
- thetabledownthere
- Theta bled own there

- But works astonishingly well in Chinese
  - 莎拉波娃现在居住在美国东南部的佛罗里达。
  - 莎拉波娃 现在 居住 在 美国 东南部 的 佛罗里达
- Modern algorithms better still:
  - probabilistic segmentation
  - Using “sequence models” like HMMs

Normalization

- Need to “normalize” terms
  - For IR, indexed text & query terms must have same form.
    - We want to match ***U.S.A.*** and ***USA***
- We most commonly implicitly define equivalence classes of terms
  - e.g., by deleting periods in a term
- Alternative is to do asymmetric expansion:
  - Enter: ***window*** Search: ***window, windows***
  - Enter: ***windows*** Search: ***Windows, windows, window***
  - Enter: ***Windows*** Search: ***Windows***
- Potentially more powerful, but less efficient

Slide from Chris Manning

Case folding

- For IR: Reduce all letters to lower case
  - exception: upper case in mid-sentence?
    - e.g., ***General Motors***
    - ***Fed*** vs. ***fed***
    - ***SAIL*** vs. ***sail***
  - Often best to lower case everything, since users will use lowercase regardless of 'correct' capitalization…
- For sentiment analysis, MT, Info extraction
  - Case is helpful ("US" versus "us" is important)

Slide from Chris Manning

Lemmatization

- Reduce inflectional/variant forms to base form
- E.g.,
  - *am, are, is* $\rightarrow$ *be*
  - *car, cars, car's, cars'* $\rightarrow$ *car*
- *the boy's cars are different colors* $\rightarrow$ *the boy car be different color*
- Lemmatization implies doing "proper" reduction to dictionary headword form

Slide from Chris Manning

Stemming

- Reduce terms to their "roots" before indexing
- "Stemming" is crude chopping of "affixes"
  - language dependent
  - e.g., ***automate(s), automatic, automation*** all reduced to ***automat***.

for example compressed and compression are both accepted as equivalent to compress.

for exampl compress and compress ar both accept as equival to compress

Slide from Chris Manning

Porter's algorithm

- Commonest algorithm for stemming English
- A sequence of phases
- each phase consists of a set of rules
  - *sses* $\rightarrow$ *ss*
  - *ies* $\rightarrow$ *i*
  - *ational* $\rightarrow$ *ate*
  - *tional* $\rightarrow$ *tion*
  - Some rules only apply to multi-syllable words
    - *(syl > 1) EMENT* $\rightarrow$ *ø*
    - *replacement* $\rightarrow$ *replac*
    - *cement* $\rightarrow$ *cement*

Slide from Chris Manning

More on Morphology

- Morphology:
  - how words are built up from smaller meaningful units called **morphemes:**
  - Stems: The core meaning bearing units
  - Affixes: Bits and pieces that adhere to stems to change their meanings and grammatical functions

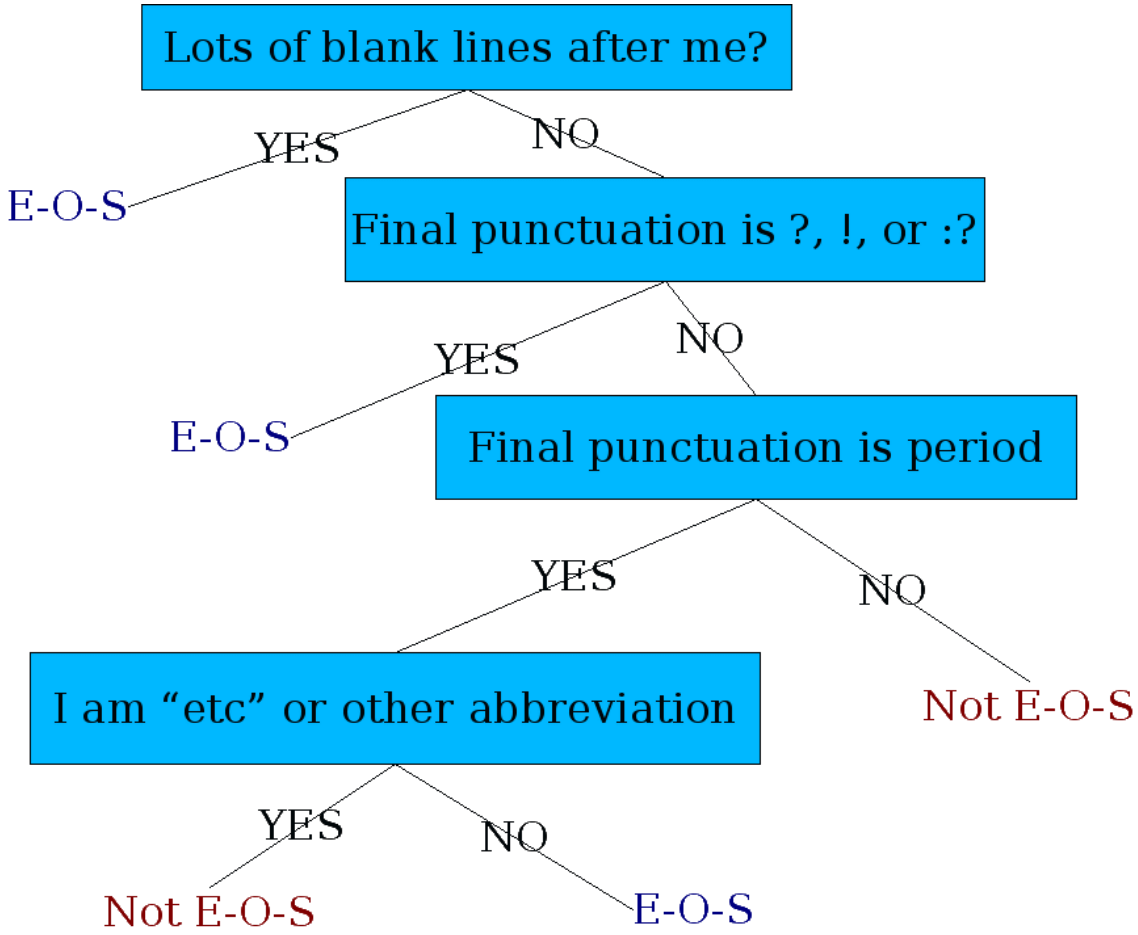
Dealing with complex morphology

- Machine translation
  - Need to know that the Spanish words quiero (‘I want’) and quieres (‘you want’) are both related to querer ‘want’
- Other languages requires segmenting morphemes
  - Turkish
  - Uygarlastiramadiklarimizdanmissinizcasina
  - ‘(behaving) as if you are among those whom we could not civilize’
  - Uygar `civilized’ + las `become’
    + tir `cause’ + ama `not able’
    + dik `past’ + lar ‘plural’
    + imiz ‘p1pl’ + dan ‘abl’
    + mis ‘past’ + siniz ‘2pl’ + casina ‘as if’

Sentence Segmentation

- !, ? relatively unambiguous
- Period "." is quite ambiguous
  - Sentence boundary
  - Abbreviations like Inc. or Dr.
- General idea:
  - Build a binary classifier:
    - Looks at a "."
    - Decides EndOfSentence/NotEOS
    - Could be hand-written rules, sequences of regular expressions, or machine-learning

Determining if a word is end-of-utterance: a Decision Tree



More sophisticated decision tree features

- Prob(word with “.” occurs at end-of-s)
- Prob(word after “.” occurs at begin-of-s)
- Length of word with “.”
- Length of word after “.”
- Case of word with “.”: Upper, Lower, Cap, Number
- Case of word after “.”: Upper, Lower, Cap, Number
- Punctuation after “.” (if any)
- Abbreviation class of word with “.” (month name, unit-of-measure, title, address name, etc)

Slide from Richard Sproat

Learning Decision Trees

- DTs are rarely built by hand
- Hand-building only possible for very simple features, domains
- Lots of algorithms for DT induction

Minimum Edit Distance

- Spell-checking
  - Non-word error detection:
    - detecting “graffe”
  - Non-word error correction:
    - figuring out that “graffe” should be “giraffe”
  - Context-dependent error detection and correction:
    - Figuring out that “war and piece” should be peace

Non-word error detection

- Any word not in a dictionary
- Assume it's a spelling error
- Need a big dictionary!

Isolated word error correction

- How do I fix “graffe”?
  - Search through all words:
    - graf
    - craft
    - grail
    - giraffe
  - Pick the one that’s closest to graffe
  - What does “closest” mean?
  - We need a distance metric.
  - The simplest one: edit distance.
    - (More sophisticated probabilistic ones: noisy channel)

Edit Distance

- The minimum edit distance between two strings
- Is the minimum number of editing operations
  - Insertion
  - Deletion
  - Substitution
- Needed to transform one into the other

Minimum Edit Distance

```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N
```

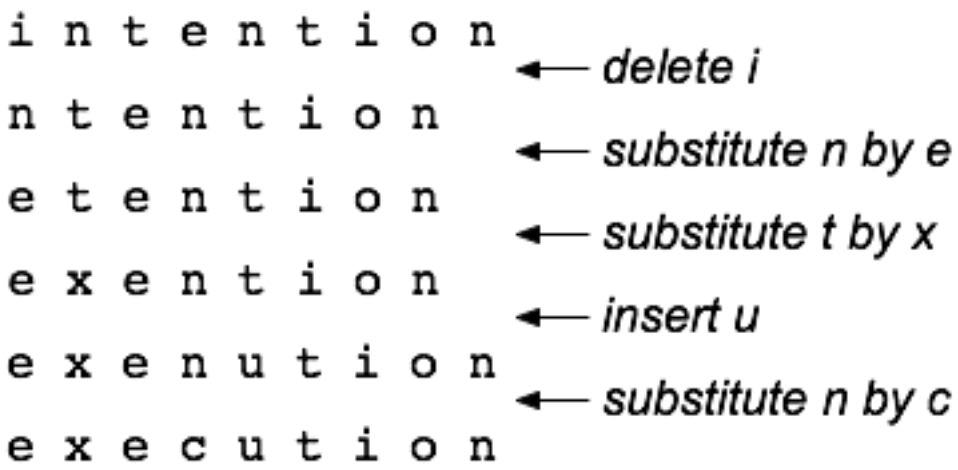
Minimum Edit Distance

```
I N T E * N T I O N
| | | | | | | | | |
* E X E C U T I O N

d s s   i s
```

- If each operation has cost of 1
  - Distance between these is 5
- If substitutions cost 2 (Levenshtein)
  - Distance between them is 8

Edit transcript



Defining Min Edit Distance

- For two strings S_1 of len n, S_2 of len m
  - distance(i,j) or D(i,j)
    - means the edit distance of S_1[1..i] and S_2[1..j]
    - i.e., the minimum number of edit operations need to transform the first i characters of S_1 into the first j characters of S_2
    - The edit distance of S_1, S_2 is D(n,m)
- We compute D(n,m) by computing D(i,j) for all i ($0 < i < n$) and j ($0 < j < m$)

Defining Min Edit Distance

- Base conditions:
  - $D(i,0) = i$
  - $D(0,j) = j$
  - Recurrence Relation:
  - $$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2 & \text{if } S_1(i) \neq S_2(j) \\ 0 & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

Dynamic Programming

- A tabular computation of $D(n,m)$
- Bottom-up
  - We compute $D(i,j)$ for small i,j
  - And compute increase $D(i,j)$ based on previously computed smaller values

The Edit Distance Table

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	**0**	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

N	9									
O	8									
I	7									
T	6									
N	5									
E	4									
T	3									
N	2									
I	1									
#	**0**	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 \\ D(i,j-1) + 1 \\ D(i-1,j-1) + \begin{cases} 2; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} \end{cases}$$

N	9	8	9	10	11	12	11	10	9	**8**
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	**0**	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Suppose we want the alignment too

- We can keep a “backtrace”
- Every time we enter a cell, remember where we came from
- Then when we reach the end, we can trace back from the upper right corner to get an alignment

Backtrace

N	9	8	9	10	11	12	11	10	9	**8**
O	8	7	8	9	10	11	10	9	8	9
I	7	6	7	8	9	10	9	8	9	10
T	6	5	6	7	8	9	8	9	10	11
N	5	4	5	6	7	8	9	10	11	10
E	4	3	4	5	6	7	8	9	10	9
T	3	4	5	6	7	8	7	8	9	8
N	2	3	4	5	6	7	8	7	8	7
I	1	2	3	4	5	6	7	6	7	8
#	**0**	1	2	3	4	5	6	7	8	9
	#	E	X	E	C	U	T	I	O	N

Adding Backtrace to MinEdit

- Base conditions:
  - D(*i*,0) = *i*
  - D(0,*j*) = *j*
- Recurrence Relation:

$$D(i,j) = \min \begin{cases} D(i-1,j) + 1 & \text{Case 1} \\ D(i,j-1) + 1 & \text{Case 2} \\ D(i-1,j-1) + \begin{cases} 1; & \text{if } S_1(i) \neq S_2(j) \\ 0; & \text{if } S_1(i) = S_2(j) \end{cases} & \text{Case 3} \end{cases}$$

$$ptr(i,j) \begin{cases} \text{LEFT} & \text{Case 1} \\ \text{DOWN} & \text{Case 2} \\ \text{DIAG} & \text{Case 3} \end{cases}$$

MinEdit with Backtrace

n	9	↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙←↓ 12	↓ 11	↓ 10	↓ 9	↙ **8**
o	8	↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↙←↓ 11	↓ 10	↓ 9	↙ **8**	← 9
i	7	↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9	↙ **8**	← 9	← 10
t	6	↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙←↓ 9	↙ **8**	← 9	← 10	←↓ 11
n	5	↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ **8**	↙←↓ 9	↙←↓ 10	↙←↓ 11	↙↓ 10
e	4	↙ 3	← 4	↙← **5**	← **6**	← 7	←↓ 8	↙←↓ 9	↙←↓ 10	↓ 9
t	3	↙←↓ 4	↙←↓ **5**	↙←↓ 6	↙←↓ 7	↙←↓ 8	↙ 7	←↓ 8	↙←↓ 9	↓ 8
n	2	↙←↓ **3**	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙←↓ 8	↓ 7	↙←↓ 8	↙ 7
i	**1**	↙←↓ 2	↙←↓ 3	↙←↓ 4	↙←↓ 5	↙←↓ 6	↙←↓ 7	↙ 6	← 7	← 8
#	**0**	1	2	3	4	5	6	7	8	9
	#	**e**	**x**	**e**	**c**	**u**	**t**	**i**	**o**	**n**

Performance

- Time:

  O(nm)

- Space:

  O(nm)

- Backtrace

  O(n+m)

Weighted Edit Distance

- Why would we add weights to the computation?
- How?

Confusion matrix

sub[X, Y] = Substitution of X (incorrect) for Y (correct)

X \ Y (correct)	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
a	0	0	7	1	342	0	0	2	118	0	1	0	0	3	76	0	0	1	35	9	9	0	1	0	5	0
b	0	0	9	9	2	2	3	1	0	0	0	5	11	5	0	10	0	0	2	1	0	0	8	0	0	0
c	6	5	0	16	0	9	5	0	0	0	1	0	7	9	1	10	2	5	39	40	1	3	7	1	1	0
d	1	10	13	0	12	0	5	5	0	0	2	3	7	3	0	1	0	43	30	22	0	0	4	0	2	0
e	388	0	3	11	0	2	2	0	89	0	0	3	0	5	93	0	0	14	12	6	15	0	1	0	18	0
f	0	15	0	3	1	0	5	2	0	0	0	3	4	1	0	0	0	6	4	12	0	0	2	0	0	0
g	4	1	11	11	9	2	0	0	0	1	1	3	0	0	2	1	3	5	13	21	0	0	1	0	3	0
h	1	8	0	3	0	0	0	0	0	0	2	0	12	14	2	3	0	3	1	11	0	0	2	0	0	0
i	103	0	0	0	146	0	1	0	0	0	0	6	0	0	49	0	0	0	2	1	47	0	2	1	15	0
j	0	1	1	9	0	0	1	0	0	0	0	2	1	0	0	0	0	0	5	0	0	0	0	0	0	0
k	1	2	8	4	1	1	2	5	0	0	0	0	5	0	2	0	0	0	6	0	0	0	4	0	0	3
l	2	10	1	4	0	4	5	6	13	0	1	0	0	14	2	5	0	11	10	2	0	0	0	0	0	0
m	1	3	7	8	0	2	0	6	0	0	4	4	0	180	0	6	0	0	9	15	13	3	2	2	3	0
n	2	7	6	5	3	0	1	19	1	0	4	35	78	0	0	7	0	28	5	7	0	0	1	2	0	2
o	91	1	1	3	116	0	0	0	25	0	2	0	0	0	0	14	0	2	4	14	39	0	0	0	18	0
p	0	11	1	2	0	6	5	0	2	9	0	2	7	6	15	0	0	1	3	6	0	4	1	0	0	0
q	0	0	1	0	0	0	27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	0	14	0	30	12	2	2	8	2	0	5	8	4	20	1	14	0	0	12	22	4	0	0	1	0	0
s	11	8	27	33	35	4	0	1	0	1	0	27	0	6	1	7	0	14	0	15	0	0	5	3	20	1
t	3	4	9	42	7	5	19	5	0	1	0	14	9	5	5	6	0	11	37	0	0	2	19	0	7	6
u	20	0	0	0	44	0	0	0	64	0	0	0	0	2	43	0	0	4	0	0	0	0	2	0	8	0
v	0	0	7	0	0	3	0	0	0	0	0	1	0	0	1	0	0	0	8	3	0	0	0	0	0	0
w	2	2	1	0	1	0	0	2	0	0	1	0	0	0	0	7	0	6	3	3	1	0	0	0	0	0
x	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	0	0	0	0	0	0	0
y	0	0	2	0	15	0	1	7	15	0	0	0	2	0	6	1	0	7	36	8	5	0	0	1	0	0
z	0	0	0	7	0	0	0	0	0	0	0	7	5	0	0	0	0	2	21	3	0	0	0	0	3	0


~ `
! 1
@ 2
# 3
$ 4
% 5
^ 6
& 7
* 8
(9
) 0
— -
+ =
Tab
Q
W
E
R
T
Y
U
I
O
P
{ [
}]
Caps Lock
A
S
D
F
G
H
J
K
L
: ;
" '
Enter
Shift
Z
X
C
V
B
N
M
< ,
> .
? /
Shift
Ctrl
Alt
Alt

Weighted Minimum Edit Distance

function MIN-EDIT-DISTANCE(*target*, *source*) **returns** *min-distance*

$n \leftarrow$ LENGTH(target)
$m \leftarrow$ LENGTH(source)
Create a distance matrix *distance*[n+1, m+1]
Initialize the zero-th row and column to the distance from the empty string
  distance[0, 0] = 0
  for each column i **from** 1 **to** n **do**
    distance [i, 0] $\leftarrow$ *distance*[i-1, 0] + *ins-cost*(*target*[i])
  for each row j **from** 1 **to** m **do**
    distance[0,j] $\leftarrow$ *distance*[0, j-1] + *del-const*(*source*[j])
for each column i **from** 1 **to** n **do**
  for each row j **from** 1 **to** m **do**
    distance[i, j] $\leftarrow$ MIN(*distance*[i-1, j] + *ins-cost*($target_{i-1}$),
                distance[i-1, j-1] + *sub-cost*($source_{j-1}$, $target_{i-1}$),
                distance[i, j-1] + *del-cost*($source_{j-1}$))
return *distance*[n, m]

Why "Dynamic Programming"

"I spent the Fall quarter (of 1950) at RAND. My first task was to find a name for multistage decision processes. An interesting question is, Where did the name, dynamic programming, come from? **The 1950s were not good years for mathematical research. We had a very interesting gentleman in Washington named Wilson. He was Secretary of Defense, and he actually had a pathological fear and hatred of the word, research**. I'm not using the term lightly; I'm using it precisely. His face would suffuse, he would turn red, and he would get violent if people used the term, research, in his presence. You can imagine how he felt, then, about the term, mathematical. The RAND Corporation was employed by the Air Force, and the Air Force had Wilson as its boss, essentially. Hence, I felt I had to do something to shield Wilson and the Air Force from the fact that I was really doing mathematics inside the RAND Corporation. What title, what name, could I choose? In the first place I was interested in planning, in decision making, in thinking. But planning, is not a good word for various reasons. **I decided therefore to use the word, "programming" I wanted to get across the idea that this was dynamic, this was multistage, this was time-varying I thought, lets kill two birds with one stone. Lets take a word that has an absolutely precise meaning, namely dynamic, in the classical physical sense. It also has a very interesting property as an adjective, and that is its impossible to use the word, dynamic, in a pejorative sense. Try thinking of some combination that will possibly give it a pejorative meaning. Its impossible. Thus, I thought dynamic programming was a good name. It was something not even a Congressman could object to**. So I used it as an umbrella for my activities."

Richard Bellman, "Eye of the Hurricane: an autobiography" 1984.

Other uses of Edit Distance in text processing

- Evaluating Machine Translation and speech recognition

```
R Spokesman confirms    senior government adviser was shot
H Spokesman said    the senior            adviser was shot dead
              S      I              D                       I
```

- Entity Extraction and Coreference
  - IBM Inc. announced today
  - IBM's profits
  - Stanford President John Hennessy announced yesterday
  - for Stanford University President John Hennessy

Summary

- Tokenization
  - Word Tokenization
  - Normalization
    - Lemmatization and stemming
  - Sentence Tokenization
- Minimum Edit Distance
  - Levenshtein distance
  - Needleman-Wunsch (weighted global alignment)
  - Smith-Waterman (local alignment)
  - Applications to:
    - spell correction, machine translation, entity extraction
    - DNA fragment combination, evolutionary similarity, mutation detection